UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/827,974	04/06/2001	Jason Souloglou		7224

36183 7590 05/06/2004

PAUL, HASTINGS, JANOFSKY & WALKER LLP
P.O. BOX 919092
SAN DIEGO, CA 92191-9092

EXAMINER	
YIGDALL, MICHAEL J	
ART UNIT	**PAPER NUMBER**
2122	

DATE MAILED: 05/06/2004

6

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/827,974	SOULOGLOU ET AL.	
	Examiner	Art Unit	
	Michael J. Yigdall	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *06 April 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-16* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-16* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *06 April 2001* is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☒ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☒ None of:

1.☒ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *2,4*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-16 are pending and have been examined. The priority date considered for the application is 10 October 1998.

Priority

2. It is noted that this application appears to claim subject matter disclosed in prior Application No. PCT/GB99/03168, filed 11 October 1999, and prior Application No. (GB) 9822075.9, filed 10 October 1998. A reference to the prior application(s) must be inserted as the first sentence of the specification of this application or in an application data sheet (37 CFR 1.76), if applicant intends to rely on the filing date of the prior application under 35 U.S.C. 119(e) or 120. See 37 CFR 1.78(a). For benefit claims under 35 U.S.C. 120, the reference must include the relationship (i.e., continuation, divisional, or continuation-in-part) of all nonprovisional applications. Also, the current status of all nonprovisional parent applications referenced should be included.

3. Acknowledgment is made of applicant's claim for foreign priority based on an application filed in the United Kingdom on 10 October 1998. It is noted, however, that applicant has not filed a certified copy of the (GB) 9822075.9 application as required by 35 U.S.C. 119(b).

Drawings

4. The drawings are objected to as failing to comply with 37 CFR 1.84(p)(4) because reference characters 1, 2 and 3 have been used to designate both tree nodes (in Figures 1-5) and intermediate representation blocks (in Figures 6 and 7). A proposed drawing correction or corrected drawings are required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

Specification

5. The following guidelines illustrate the preferred layout for the specification of a utility application. These guidelines are suggested for the applicant's use.

Arrangement of the Specification:

As provided in 37 CFR 1.77(b), the specification of a utility application should include the following sections in order. Each of the lettered items should appear in upper case, without underlining or bold type, as a section heading. If no text follows the section heading, the phrase "Not Applicable" should follow the section heading:

(a) TITLE OF THE INVENTION.

(b) CROSS-REFERENCE TO RELATED APPLICATIONS.

(c) STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT.

(d) INCORPORATION-BY-REFERENCE OF MATERIAL SUBMITTED ON A COMPACT DISC (See 37 CFR 1.52(e)(5) and MPEP 608.05. Computer program listings (37 CFR 1.96(c)), "Sequence Listings" (37 CFR 1.821(c)), and tables having more than 50 pages of text are permitted to be submitted on compact discs.) or
REFERENCE TO A "MICROFICHE APPENDIX" (See MPEP § 608.05(a). "Microfiche Appendices" were accepted by the Office until March 1, 2001.)

(e) BACKGROUND OF THE INVENTION.
(1) Field of the Invention.
(2) Description of Related Art including information disclosed under 37 CFR 1.97 and 1.98.

(f) BRIEF SUMMARY OF THE INVENTION.

(g) BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWING(S).

(h) DETAILED DESCRIPTION OF THE INVENTION.

(i) CLAIM OR CLAIMS (commencing on a separate sheet).

(j) ABSTRACT OF THE DISCLOSURE (commencing on a separate sheet).

(k) SEQUENCE LISTING (See MPEP § 2424 and 37 CFR 1.821-1.825. A "Sequence Listing" is required on paper if the application discloses a nucleotide or amino acid sequence as defined in 37 CFR 1.821(a) and if the required "Sequence Listing" is not submitted as an electronic document on compact disc).

Double Patenting

6. A rejection based on double patenting of the "same invention" type finds its support in the language of 35 U.S.C. 101 which states that "whoever invents or discovers any new and useful process ... may obtain a patent therefor ..." (emphasis added). Thus, the term "same invention," in this context, means an invention drawn to identical subject matter. See *Miller v. Eagle Mfg. Co.*, 151 U.S. 186 (1894); *In re Ockert*, 245 F.2d 467, 114 USPQ 330 (CCPA 1957); and *In re Vogel*, 422 F.2d 438, 164 USPQ 619 (CCPA 1970).

A statutory type (35 U.S.C. 101) double patenting rejection can be overcome by canceling or amending the conflicting claims so they are no longer coextensive in scope. The filing of a terminal disclaimer cannot overcome a double patenting rejection based upon 35 U.S.C. 101.

7. Claims 1-16 are provisionally rejected under 35 U.S.C. 101 as claiming the same invention as that of claims 1-16 of copending Application No. 10/164,789. This is a provisional double patenting rejection since the conflicting claims have not in fact been patented.

8. Claims 1-16 are provisionally rejected under 35 U.S.C. 101 as claiming the same invention as that of claims 1-16 of copending Application No. 10/165,457. This is a provisional double patenting rejection since the conflicting claims have not in fact been patented.

Claim Rejections - 35 USC § 102

9. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

10. Claim 13 is rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Pat. No. 5,768,593 to Walters et al. (hereinafter "Walters").

With respect to claim 13, Walters discloses a method of generating target code representation of program code (see the title and abstract), the method comprising the computer implemented steps of:

on an initial translation of a given portion of the program code, generating and storing only target code which is required to execute that portion of program code with a prevailing set of conditions (see column 7, lines 52-63, which shows generating a block of native or target code for execution on an initial translation); and

whenever subsequently the same portion of program code is entered, determining whether target code has previously been generated and stored for that portion of program code for the subsequent conditions, and if no such target code has previously been generated, generating additional target code required to execute said portion of program code with said subsequent conditions (see column 7, lines 16-23 and 52-63, which shows determining whether a

block of code has been generated and stored in a table, and if it has not, subsequently generating that code).

Claim Rejections - 35 USC § 103

11. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

12. Claims 1-5, 7-12, 15 and 16 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pat. No. 5,613,117 to Davidson et al. (hereinafter "Davidson") in view of Walters.

With respect to claim 1, Davidson discloses a method of generating an intermediate representation of program code (see the title and abstract), the method comprising the computer implemented steps of:

on an initial translation of a given portion of program code, generating and storing only intermediate representation which is required to execute that portion of program code with a prevailing set of conditions (see column 8, line 63 to column 9, line 2, which shows a first translation stage for generating an intermediate language representation of program source code, and column 11, lines 10-26, which shows generating the representation one node at a time).

Although Davidson discloses storing data structures for the intermediate representation and a symbol table (see column 27, lines 8-21), Davidson does not expressly disclose the step of:

whenever subsequently the same portion of program code is entered, determining whether intermediate representation has previously been generated and stored for that portion of program code for the subsequent conditions, and if no such intermediate representation has previously been generated, generating additional intermediate representation required to execute said portion of program code with said subsequent conditions.

However, Walters discloses a cross-compiler (see the title and abstract) comprising the step of determining whether a code block has previously been translated and stored in a table, and if it has not, subsequently translating that portion of the code to be executed (see column 7, lines 16-23 and 52-63). This code caching feature increases efficiency and enables previously translated code to be reused (see column 4, lines 46-67).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the code generation method of Davidson with the code caching feature taught by Walters, for the purpose of increasing efficiency and enabling the reuse of previously generated code.

With respect to claim 2, the combination of Davidson and Walters further discloses the limitations wherein the conditions are entry conditions (see Davidson, column 7, line 66 to column 8, line 8, which shows identifying blocks based on entry conditions), and the method comprises the computer implemented steps of:

generating an Intermediate Representation Block (IR Block) of intermediate representation for each Basic Block of program code as it is required by the program, each IR Block representing a respective Basic Block of program code for a particular entry condition (see Davidson, column 8, line 63 to column 9, line 2, which shows generating intermediate language

blocks from program source code, and column 7, line 66 to column 8, line 8, which shows that the blocks are associated with particular entry conditions);

storing target code corresponding to each IR Block (see Davidson, column 12, lines 22-29, which shows storing target machine code based on the intermediate representation); and

when the program requires execution of a Basic Block for a given entry condition, either:

(a) if there is stored target code representing that Basic Block for that given entry condition, using said stored target code (see Walters, column 7, lines 16-23, which shows using stored code if it is found for that entry condition); or

(b) if there is no stored target code representing that Basic Block for that given entry condition, generating a further IR Block representative of that Basic Block for that given entry condition (see Walters, column 7, lines 52-63, which shows generating code if it is not found for that entry condition).

As set forth above, it would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the code generation method of Davidson with the code caching feature taught by Walters, for the purpose of increasing efficiency and enabling the reuse of previously generated code.

With respect to claim 3, the combination of Davidson and Walters further discloses the limitation wherein the intermediate representation of the program code is generated dynamically as the program code is running (see Walters, column 3, lines 35-53, which shows that the code is generated at run time), the method comprising the computer implemented steps of:

at a first iteration of a particular subject code instruction having a plurality of possible effects or functions, generating and storing special-case intermediate representation representing

only the specific functionality required at that iteration (see Davidson, column 7, lines 24-65, which shows generating the intermediate representation in terms of tuples that represent the functionality of an instruction; see also column 8, lines 9-17, which shows that the tuples represent the effects of an instruction); and

at each subsequent iteration of the same subject code instruction, determining whether special-case intermediate representation has been generated for the required functionality required at said subsequent iteration and generating additional special-case intermediate representation specific to that functionality if no such special-case intermediate representation has previously been generated (see Walters, column 7, lines 16-23 and 52-63, which shows determining whether code has been generated and stored in a table, and if it has not, subsequently generating that code).

As set forth above, it would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the code generation method of Davidson with the code caching feature taught by Walters, for the purpose of increasing efficiency and enabling the reuse of previously generated code.

Furthermore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to perform the code generation method of Davidson dynamically as the program code is running, as taught by Walters, for the purpose of improving execution time (see Walters, column 2, lines 17-23).

With respect to claim 4, the combination of Davidson and Walters further discloses the limitation wherein the said special-case intermediate representation is generated and stored an associated test procedure is generated and stored to determine on subsequent iterations of the

respective subject code instruction whether the required functionality is the same as that represented by the associated stored special-case intermediate representation (see Davidson, column 13, lines 7-44, which shows using a test procedure to determine whether an instruction has the same effect or functionality as another instruction), and where additional special-case intermediate representation is required an additional test procedure associated with that special-case intermediate representation is generated and stored with that additional special-case intermediate representation (see column 13, line 60 to column 14, line 17, which shows generating additional classes or procedures to determine the effects of instructions).

With respect to claim 5, the combination of Davidson and Walters further discloses the limitation wherein the additional special case intermediate representation for a particular subject code instruction and the additional associated test procedure is stored at least initially in subordinate relation to any existing special-case intermediate representation and associated test procedures stored to represent the same subject instruction (see Davidson, column 10, lines 40-44, which shows that the intermediate representation is stored as a graph of linked nodes, i.e. the nodes are stored in subordinate relation to one another), such that upon the second and subsequent iteration of a subject code instruction determination of whether or not required special-case intermediate representation has previously been generated is made by performing said test procedures in the order in which they were generated and stored until either it is determined that special-case intermediate representation of the required functionality exists or it is determined that no such required special-case intermediate representation exists in which case more additional intermediate representation and another associated test procedure is generated; (see column 14, lines 21-52, which shows performing the test procedures in order based on the

flow paths between blocks, i.e. in the order the intermediate representation would have been generated and stored; see also column 16, lines 27-46, which further shows determining the order of the blocks in conjunction with the test procedures).

With respect to claim 7, the combination of Davidson and Walters further discloses translating the program code written for execution by a processor of a first type so that the program code may be executed by a processor of a second type, using the generated intermediate representation (see Walters, column 3, lines 35-53, which shows translating non-native code into native code for execution).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the intermediate representation generated by Davidson for program code translation as taught by Walters, in order to enable the non-native code to be executed by the native processor.

With respect to claim 8, the combination of Davidson and Walters further discloses the limitation wherein said translation is dynamic and performed as the program code is run (see Walters, column 3, lines 35-53, which further shows that the code is translated at run time).

With respect to claim 9, the combination of Davidson and Walters further discloses optimising the program code by optimising said intermediate representation (see Davidson, column 22, lines 13-33, which shows optimizing the program code using the intermediate representation).

With respect to claim 10, the combination of Davidson and Walters further discloses the limitation wherein the method is used to optimise the program code written for execution by a processor of a first type so that the program code may be executed more efficiently by that processor (see Davidson, column 22, lines 13-33, which shows optimizing the program code using the intermediate representation, and Walters, column 3, lines 35-53, which shows translating non-native code into native code for execution).

With respect to claim 11, Davidson discloses a method for generating an intermediate representation of program code written for running on a programmable machine (see the title and abstract), said method comprising:

(i) generating a plurality of register objects for holding variable values to be generated by the program code (see column 7, lines 24-65, which shows generating the intermediate representation in terms of tuples, and column 33, lines 41-47, which shows that the tuples may serve as register objects); and

(ii) generating a plurality of expression objects representing fixed values and/or relationships between said fixed values and said variable values according to said program code (see column 7, lines 24-65, which shows generating the intermediate representation in terms of tuples that serve as expression objects);

Although Davidson discloses generating intermediate language blocks from program source code (see column 8, line 63 to column 9, line 2) and storing the associated data structures and a symbol table (see column 27, lines 8-21), Davidson does not expressly disclose the limitation wherein said intermediate representation is generated and stored for a block of program code and subsequently re-used if the same block of program code is later re-entered.

However, Walters discloses a cross-compiler (see the title and abstract) wherein translated code blocks are stored in a table and subsequently reused if the blocks are later reentered (see column 7, lines 16-23). By enabling previously translated code to be reused, this code-caching feature increases the efficiency of the translation (see column 4, lines 46-67).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the code generation method of Davidson with the code caching feature taught by Walters, for the purpose of increasing efficiency and enabling the reuse of previously generated code.

Davidson further discloses the limitation wherein at least one block of program code can have alternative un-used entry conditions or effects or functions and said intermediate representation is only initially generated and stored as required to execute that block of program code with a then prevailing set of conditions (see column 7, line 66 to column 8, line 8, which shows that the blocks are associated with particular entry conditions, and column 8, lines 9-17, which shows that the tuples used in the intermediate representation represent the effects or functions of an instruction).

With respect to claim 12, the combination of Davidson and Walters further discloses the limitation wherein for a given block of program code, it is determined whether a previously stored intermediate representation therefor was for the same now currently prevailing set of conditions and, if not, then generating and storing additional intermediate representation as required to execute the block of program code for the new now currently prevailing set of conditions (see Walters, column 7, lines 16-23 and 52-63, which shows determining whether

code has been generated and stored in a table, and if it has not, subsequently generating that code).

As set forth above, it would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the code generation method of Davidson with the code caching feature taught by Walters, for the purpose of increasing efficiency and enabling the reuse of previously generated code.

With respect to claim 15, see the explanation for claim 1 set forth above. The system recited in claim 15 is analogous to the method of claim 1. Note that Davidson further discloses a system and the means for performing the recited method (see the abstract).

With respect to claim 16, see the explanation for claim 11 set forth above. The system recited in claim 16 is analogous to the method of claim 11. Note that Davidson further discloses a system and the means for performing the recited method (see the abstract).

13. Claim 6 is rejected under 35 U.S.C. 103(a) as being unpatentable over Davidson and Walters, as applied to claim 5 above, and further in view of U.S. Pat. No. 6,631,514 to Le.

With respect to claim 6, although the combination of Davidson and Walters discloses optimizing the intermediate representation with code motion (see Davidson, column 3, lines 53-56) and using test procedures (see column 3, lines 56-63), the combination of Davidson and Walters does not expressly disclose the limitation wherein the intermediate representation is optimised by adjusting the ordering of the test procedures such that test procedures associated with more frequently used special-case intermediate representation are run before test procedures

associated with less frequently used special-case intermediate representation rather than ordering the test procedures in the order in which they are generated.

However, Le discloses a translation system wherein the program code is optimized by reordering the instructions (see the title and abstract), for the purpose of achieving higher performance (see column 3, lines 13-22).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the code generation method of Davidson and Walters with the reordering feature taught by Le, so that more frequently used blocks and/or test procedures may be run first, thereby achieving higher performance.

14. Claims 14 is rejected under 35 U.S.C. 103(a) as being unpatentable over Walters in view of Davidson.

With respect to claim 14, Walters discloses a method of dynamically translating first computer program code written for compilation and/or translation and running on a first programmable machine into second computer program code for running on a different second programmable machine (see the title and abstract).

Although Walters discloses translating a block of said first computer program code into a block of said second computer program code (see column 3, lines 35-44), Walters does not expressly disclose:

(a) generating an intermediate representation of a block of said first computer program code;

(b) generating a block of said second computer program code from said intermediate representation;

However, Davidson discloses generating an intermediate representation of program source code in order to enable language-independent optimizations (see column 3, line 22 to column 4, line 2).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to generate an intermediate representation, as taught by Davidson, during the translation process of Walters, for the purpose of optimizing the program code regardless of the programming language.

Walters further discloses:

(c) running said block of second computer program code on said second programmable machine (see column 3, lines 35-53, which shows that the code is translated at run time and executed on the second processor), and

(d) repeating steps a-c in real time for at least the blocks of first computer program code needed for a current emulated execution of the first computer program code on said second programmable machine (see column 3, lines 35-53, which shows that the cross-compiler is operated in real time as the program is running).

Conclusion

15. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. U.S. Pat. No. 5,586,323 to Koizumi et al. discloses a program translation system comprising an intermediate representation and abstract registers.

16. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Michael J. Yigdall whose telephone number is (703) 305-0352. The examiner can normally be reached on Monday through Friday from 7:30am to 4:00pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (703) 305-4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

MY

Michael J. Yigdall
Examiner
Art Unit 2122

mjy
May 3, 2004

TUAN DAM
SUPERVISORY PATENT EXAMINER